WEST Search History

DATE: Friday, July 11, 2003

Set Name side by side	Query	Hit Count	Set Name result set
DB=EPAB,DWPI,TDBD; PLUR=YES; OP=ADJ			
L12	L6 and program$ intent	0	L12
DB=USPT,PGPB; PLUR=YES; OP=ADJ			
L11	L4 and program$ intent	6	L11
L10	L4 and program intent	0	L10
L9	L4 and intent	33	L9
DB=EPAB,DWPI,TDBD; PLUR=YES; OP=ADJ			
L8	L6 and l5	0	L8
L7	L6 and l6	1358	L7
L6	identif$ near2 node	1358	L6
L5	program$ near construct	92	L5
DB=USPT,PGPB; PLUR=YES; OP=ADJ			
L4	L3 and (identif$ near2 node)	75	L4
L3	L2 and node and tree	241	L3
L2	program$ near construct	1284	L2
L1	(((717/141 |717/142 |717/143 |717/144 |717/145 |717/146 |717/147 |717/117)!.CCLS.))	683	L1

END OF SEARCH HISTORY

Search Results - Record(s) 21 through 33 of 33 returned.

☐ 21. Document ID: US 6339832 B1

L9: Entry 21 of 33 File: USPT Jan 15, 2002

US-PAT-NO: 6339832
DOCUMENT-IDENTIFIER: US 6339832 B1

TITLE: Exception response table in environment services patterns

DATE-ISSUED: January 15, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bowman-Amuah; Michel K.	Colorado Springs	CO		

US-CL-CURRENT: 714/35; 710/266, 710/48, 714/50

ABSTRACT:

A system, method and article of manufacture are provided for recording exception handling requirements for maintaining a consistent error handling approach. An exception response table is provided in which an exception is recorded. The context of the exception is entered in the exception response table and a response for the exception is listed in the exception response table. The response is subsequently outputted upon the exception occurring in the context.

18 Claims, 195 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 123

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw Desc | Image

☐ 22. Document ID: US 6332163 B1

L9: Entry 22 of 33 File: USPT Dec 18, 2001

US-PAT-NO: 6332163
DOCUMENT-IDENTIFIER: US 6332163 B1

TITLE: Method for providing communication services over a computer network system

DATE-ISSUED: December 18, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bowman-Amuah; Michel K.	Colorado Springs	CO		

US-CL-CURRENT: 709/231; 709/217, 709/223, 709/227, 709/329

ABSTRACT:

A system, method and article of manufacture are provided for implementing communication services patterns. A fixed format stream-based communication system is provided and service is delivered via a globally addressable interface. Access is afforded to a legacy system. Service is delivered via a locally addressable interface. A null value is communicated and data is transmitted from a server to a client via pages. A naming service and a client are interfaced with the naming service allowing access to a plurality of different sets of services from a plurality of globally addressable interfaces. A stream-based communication system is provided and data is efficiently retrieved.

15 Claims, 195 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 123

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw Desc | Image

---

☐ 23. Document ID: US 6289382 B1

L9: Entry 23 of 33 File: USPT Sep 11, 2001

US-PAT-NO: 6289382
DOCUMENT-IDENTIFIER: US 6289382 B1

TITLE: System, method and article of manufacture for a globally addressable interface in a communication services patterns environment

DATE-ISSUED: September 11, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bowman-Amuah; Michel K.	Colorado Springs	CO		

US-CL-CURRENT: 709/226

ABSTRACT:

A system, method, and article of manufacture are provided for delivering service via a globally addressable interface. A plurality of interfaces are provided with access allowed to a plurality of different sets of services from each of the interfaces. Each interface has a unique set of services associated therewith. Each of the interfaces is named with a name indicative of the unique set of services associated therewith. The names of the interfaces are then broadcast to a plurality of systems requiring service.

15 Claims, 195 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 122

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw Desc | Image

---

☐ 24. Document ID: US 6189143 B1

L9: Entry 24 of 33 File: USPT Feb 13, 2001

US-PAT-NO: 6189143
DOCUMENT-IDENTIFIER: US 6189143 B1

TITLE: Method and system for reducing an intentional program tree represented by high-level computational constructs

.DATE-ISSUED: February [redacted], 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Simonyi; Charles	Medina	WA		

US-CL-CURRENT: 717/109; 717/144

ABSTRACT:

A method and system is described for generating executable code for a computer program, A programmer creates an intentional program tree using a syntax-independent editor. The editor allows a programmer to directly manipulate the intentional program tree. The intentional program tree has nodes. Each node represents a high-level computational construct of the computer program. For each node representing a high-level computational construct, the system transforms the node into an implementation of the high-level computational construct using low-level computational constructs. For each node representing a low-level computational construct, the system generates executable code that implements the low-level computational construct. The system further provides that where a high-level computational construct has a plurality of implementations of the high-level computational construct, the system transforms the nodes by selecting one of the implementations and transforms the node in accordance with the selected implementation. The system further provides that the implementation is selected by automatically analyzing semantics of the intentional program tree.

64 Claims, 38 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 35

☐ 25. Document ID: US 6097888 A

L9: Entry 25 of 33 File: USPT Aug 1, 2000

US-PAT-NO: 6097888
DOCUMENT-IDENTIFIER: US 6097888 A

TITLE: Method and system for reducing an intentional program tree represented by high-level computational constructs

DATE-ISSUED: August 1, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Simonyi; Charles	Medina	WA		

US-CL-CURRENT: 717/144; 345/967, 717/146

ABSTRACT:

A method and system for generating a computer program in the manner that uses no computer programming language syntax. The system represents a computer program as an intentional program tree, which is a high-level program tree that is a syntax-independent representation using high-level computational constructs. The intentional program tree represents a programmer's intent, rather than an implementation of the programmer's intent. The programmer creates an intentional program tree using a syntax-independent editor. The editors allows a programmer to directly

manipulate the intentional program tree. Because the program is stored as an intentional program tree in a syntax-independent manner, the editor allows the

program to select in which of a various programming language the computer program is to be displayed. In addition, the system transforms an intentional program tree to a reduced program tree, which is a program tree comprising low-level computational constructs, in a process called reduction. The reduction process replaces expressions of programmer's intents with a representation of one of possible multiple implementations of those intents using low-level computational constructs.

55 Claims, 38 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 35

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments KWIC | Draw Desc | Image

---

☐ 26. Document ID: US 6078746 A

L9: Entry 26 of 33 File: USPT Jun 20, 2000

US-PAT-NO: 6078746
DOCUMENT-IDENTIFIER: US 6078746 A

TITLE: Method and system for reducing an intentional program tree represented by high-level computational constructs

DATE-ISSUED: June 20, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Simonyi; Charles	Medina	WA		

US-CL-CURRENT: 717/144; 717/114

ABSTRACT:

A method and system for generating a computer program in the manner that uses no computer programming language syntax. The system represents a computer program as an intentional program tree, which is a high-level program tree that is a syntax-independent representation using high-level computational constructs. The intentional program tree represents a programmer's intent, rather than an implementation of the programmer's intent. The programmer creates an intentional program tree using a syntax-independent editor. The editors allows a programmer to directly manipulate the intentional program tree. Because the program is stored as an intentional program tree in a syntax-independent manner, the editor allows the program to select in which of a various programming language the computer program is to be displayed. In addition, the system transforms an intentional program tree to a reduced program tree, which is a program tree comprising low-level computational constructs, in a process called reduction. The reduction process replaces expressions of programmer's intents with a representation of one of possible multiple implementations of those intents using low-level computational constructs.

16 Claims, 38 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 35

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments KWIC | Draw Desc | Image

---

☐ 27. Document ID: US 6070007 A

L9: Entry 27 of 33 File: USPT May 30, 2000

US-PAT-NO: 6070007

DOCUMENT-IDENTIFIER: US 6070007 A

TITLE: Method and system for reducing an intentional program tree represented by high-level computational constructs

DATE-ISSUED: May 30, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Simonyi; Charles	Medina	WA		

US-CL-CURRENT: 717/106; 717/127

ABSTRACT:

A method and system for generating a computer program in the manner that uses no computer programming language syntax. The system represents a computer program as an intentional program tree, which is a high-level program tree that is a syntax-independent representation using high-level computational constructs. The intentional program tree represents a programmer's intent, rather than an implementation of the programmer's intent. The programmer creates an intentional program tree using a syntax-independent editor. The editors allows a programmer to directly manipulate the intentional program tree. Because the program is stored as an intentional program tree in a syntax-independent manner, the editor allows the program to select in which of a various programming language the computer program is to be displayed. In addition, the system transforms an intentional program tree to a reduced program tree, which is a program tree comprising low-level computational constructs, in a process called reduction. The reduction process replaces expressions of programmer's intents with a representation of one of possible multiple implementations of those intents using low-level computational constructs.

72 Claims, 38 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 35

☐ 28. Document ID: US 5911072 A

L9: Entry 28 of 33 File: USPT Jun 8, 1999

US-PAT-NO: 5911072
DOCUMENT-IDENTIFIER: US 5911072 A

TITLE: Method and system for reducing an intentional program tree represented by high-level computational constructs

DATE-ISSUED: June 8, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Simonyi; Charles	Medina	WA		

US-CL-CURRENT: 717/105; 717/113

ABSTRACT:

A method and system for generating a computer program in the manner that uses no computer programming language syntax. The system represents a computer program as an intentional program tree, which is a high-level program tree that is a syntax-independent representation using high-level computational constructs. The intentional program tree represents a programmer's intent, rather than an implementation of the programmer's intent. The programmer creates an intentional

program tree using a syntax-independent editor. The editors allows a programmer to directly manipulate the intentional program tree. Because the program is stored as an intentional program tree in a syntax-independent manner, the editor allows the program to select in which of a various programming language the computer program is to be displayed. In addition, the system transforms an intentional program tree to a reduced program tree, which is a program tree comprising low-level computational constructs, in a process called reduction. The reduction process replaces expressions of programmer's intents with a representation of one of possible multiple implementations of those intents using low-level computational constructs.

28 Claims, 38 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 35

☐ 29. Document ID: US 5822593 A

L9: Entry 29 of 33 File: USPT Oct 13, 1998

US-PAT-NO: 5822593
DOCUMENT-IDENTIFIER: US 5822593 A

TITLE: High-level loop fusion

DATE-ISSUED: October 13, 1998

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Lamping; John O.	Los Altos	CA		
Kiczales; Gregor J.	Palo Alto	CA		
Mendhekar; Anurag D.	Sunnyvale	CA		

US-CL-CURRENT: 717/161; 717/144, 717/156

ABSTRACT:

A processor is provided with a software program specifying an overall computation that includes operations. Each operation implies a set of subcomputations, without explicitly specifying a control structure for carrying out the subcomputations according to a particular sequencing. The operations include a first and a second operation, and the provided software program further specifies how the first and second operations are combined in the overall computation. For example, the first and second operations can each imply, respectively, a first and a second computational loop, the first loop including the subcomputations of the first operation, the second loop including the subcomputations of the second operation. A description of possible sequencings of subcomputations of the first and second operations is provided, to be used in implementing the specified combination of the first and second operations, the description including a set of constraints on the sequencing of subcomputations of the first and second operations. A software program is automatically generated that includes a combined operation implementing the specified combination of the first and second operations. The combined operation has a control structure for carrying out the subcomputations of the first and second operations in accordance with the constraints. This control structure can be, for example, a computational loop. If the first and second operations imply, respectively, first and second computational loops, the control structure of the combined operation can be, for example, a computational loop including a fusion of the first and second loops.

28 Claims, 11 Drawing figures
Exemplary Claim Number: 28
Number of Drawing Sheets: 9

☐ 30. Document ID: US 5528508 A

L9: Entry 30 of 33 File: USPT Jun 18, 1996

US-PAT-NO: 5528508
DOCUMENT-IDENTIFIER: US 5528508 A

TITLE: System and method for verifying a hierarchical circuit design

DATE-ISSUED: June 18, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Russell; Philip J.	Alresford			GB
Weinert; Glenwood S.	San Jose	CA		

US-CL-CURRENT: 716/8; 716/11, 716/5

ABSTRACT:

A computer-based system and method is provided for building a representation of a hierarchical circuit design and component intrusions for the components making up the circuit design, as well as for verifying a design so-represented. For a subject hierarchical circuit design, a VLSI circuit design component representing a leaf design entity is isolated. A set of locations in the design where the component appears is determined. These locations represent unique instances of the leaf design entity. A set of links is associated with the VLSI circuit design component and the locations. The links connect various ones of the locations to one another to denote placement of the component within the hierarchical circuit design. To complete the representation, a set of instance counts is computed, one instance count for each location in the design where the component is represented. Each instance count denotes the number of instances of the component represented at the location with which the instance count is associated. Additional features of the invention include applicability to numerous types of design components (e.g., devices, nets, microprocessors, resistors), correspondence between each node of the inverse layout graph and a unique placement in the hierarchical circuit design, and the ability to determine intrusions according to any measure of proximity.

26 Claims, 80 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 56

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

☐ 31. Document ID: US 5519628 A

L9: Entry 31 of 33 File: USPT May 21, 1996

US-PAT-NO: 5519628
DOCUMENT-IDENTIFIER: US 5519628 A

TITLE: System and method for formulating subsets of a hierarchical circuit design

DATE-ISSUED: May 21, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Russell; Philip J.	Alresford			GB
Weinert; Glenwood S.	San Jose	CA		

US-CL-CURRENT: 716/10

ABSTRACT:

A computer-based system and method is provided for building subsets of a hierarchical circuit design. A VLSI circuit design component is stored in computer memory. The design component identifies a leaf design entity in the hierarchical circuit design. A set of placements is determined representing positions in the hierarchical circuit design where the VLSI circuit design component appears. The placements form a subset of instances of the leaf design entity. A set of links is created. The links are associated in memory with both the VLSI circuit design component and the placements, and connect various ones of the placements to one another to further denote placement of the VLSI circuit design component within the hierarchical circuit design. A subset list is appended to the VLSI circuit design component in computer memory. The subset list denotes the previously-determined subset and includes placements where the VLSI circuit design component is identified in the hierarchical circuit design. The identified placements may indicate exclusion of a particular instance of a design component from the hierarchical circuit design, or inclusion of a particular instance.

17 Claims, 80 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 56

☐ 32. Document ID: US 5497334 A

L9: Entry 32 of 33 File: USPT Mar 5, 1996

US-PAT-NO: 5497334
DOCUMENT-IDENTIFIER: US 5497334 A

TITLE: Application generator for use in verifying a hierarchical circuit design

DATE-ISSUED: March 5, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Russell; Philip J.	Alresford			GB
Weinert; Glenwood S.	San Jose	CA		

US-CL-CURRENT: 716/5

ABSTRACT:

A computer based system and method is provided for generating a design verification scheme for a hierarchical circuit design. A set of directives received describing design checks to be performed on a hierarchical circuit design. The directives are functionally decomposed into primitive functions required to perform them. A primary iteration level is established for each directive, and a data flow dependency is determined for the directives. Based on the data now dependency, a sequence or operations is organized. The operations are optimized in one or more ways to improve the efficiency of the design verification process. The optimized operations are coded into an application program which executes in a computer processor. The application program accesses the VLSI circuit design under review and performs the directives using the data structures allocated during schema generation.

14 Claims, 80 Drawing Figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 56

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments KWIC | Draw Desc | Image

☐ 33. Document ID: US 5481473 A

L9: Entry 33 of 33 File: USPT Jan 2, 1996

US-PAT-NO: 5481473
DOCUMENT-IDENTIFIER: US 5481473 A

TITLE: System and method for building interconnections in a hierarchical circuit design

DATE-ISSUED: January 2, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Kim; Young O.	San Jose	CA		
Russell; Philip J.	Alresford			GB
Weinert; Glenwood S.	San Jose	CA		

US-CL-CURRENT: 716/5; 716/12

ABSTRACT:

A computer-based system and method is provided for creating a representation of interconnections between VLSI circuit design components. A VLSI circuit design component identifying a leaf design entity is stored in memory. Placements in the design where the design component appears are stored in memory. A set of links is formed to connect placements to one another. The links further specify placement of the design component in the circuit design. The interconnections themselves are then computed. The interconnections denote where placements of the VLSI circuit design component instances are interconnected, and may specify any meaningful coupling, such as electrical conductivity, magnetic, or optical. The interconnections are represented by a nested net graph which includes a list of nets, and instance counts associated with the nets. The nested net graph may also include a second list, which specifies instances of lower nested nets contained in the nested net graph. The nested net graph may further include a shape-to-net table attached at the root of the nested net graph. The shape-to-net table defines a mapping from the VLSI circuit design component to a corresponding net. Also provided is a system and method for building interconnections using a bridge component, or bridge net. The bridge net denotes the interconnection between two nets derived from a pair of VLSI circuit design component instances.

18 Claims, 80 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 56

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments KWIC | Draw Desc | Image

Generate Collection | Print

Terms	Documents
L4 and intent	33

CiteSeer Find: programming intent computational co Documents Citations

Searching for **PHRASE programming intent computational construct node tree**.
Restrict to: Header Title Order by: Citations Hubs Usage Date Try: Amazon B&N Google (RI) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. **Only retrieving 125 documents (System busy - maximum reduced).** Retrieving documents... documents... **Order: relevance to query.**

Evolving Recusive Programs for Tree Search - Scott Brave (Correct)
Scott Brave This article compares basic genetic **programming**, genetic **programming** with automatically
of these techniques is also investigated. The **computational** effort required to reach a solution using
is doing anything but simply looping at a single **node** in the **tree**, it will search and reach the base
brave.www.media.mit.edu/people/brave/publications/AiGPII.ps

Prioritization in Parallel Symbolic Computing - Kale, Ramkumar, Saletore, Sinha (1993) (Correct) (5 citations)
been implemented in the Charm portable parallel **programming** system. Performance results on shared-memory
we explored the 15 puzzle problem further by **intent**ionally weakening the heuristic. The heuristic
sets itself an ambitious goal: that of building **computational** systems that are capable of intelligent
nscp.upenn.edu/parallel/environments/charm/papers/Symbolic_LNCS93.ps.gz

Design and Implementation of a General Purpose Parallel.. - Mark Chu-Carroll (Correct)
and Implementation of a General Purpose Parallel **Programming** System Mark Chu-Carroll (presenting author)
of scientific computation, there are many **computationally** intensive applications that run slowly.
an example of a language extended with composite **constructs**. Finally, we describe the compiler technology
www.eecis.udel.edu/~pollock/papers/hpcn.ps

Localized Multicast Routing - Shaikh, Lu, Shin (1995) (Correct)
that the source uses global cost information to **construct** a multicast **tree**. Our algorithm does not
it uses cost information only from neighboring **nodes** as it proceeds which makes it more practical from
global cost information to **construct** a multicast **tree**. Our algorithm does not require the use of global
www.eecs.umich.edu/~ashaikh/research/papers/gcomm95.ps.Z

Encoding Lexicalized Tree Adjoining Grammars with a.. - Evans, Gazdar, Weir (1995) (Correct) (7 citations)
available, on various platforms and **programming** languages. 5 In fact, LTAG commonly
Gibbon. 1992. ILEX: a linguistic approach to **computational** lexica. In Ursula Klenk, ed.Computatio
relations, including unbounded dependency **construct**ions, are represented lexically and are thus
ftp.itri.bton.ac.uk/pub/reports/ITRI-95-07.ps.gz

Dynamic Load Balancing of Unstructured Computations in .. - Srivastava, Han.. (1998) (Correct) (1 citation)
the three parallel formulations using the MPI **programming** library. We use binary split- ting at each
large. Hence, it is highly desirable to design **computationally** efficient as well as scalable algorithms.
service plans. Highly parallel algorithms for **construct**ing classification decision **trees** are desirable
ftp.cise.ufl.edu/pub/faculty/ranka/Proceedings/p9.ps

Investigating Parallel Interpretation-Tree Model.. - Proset-Linda Hasselbring (1994) (Correct)
:2 2.2 Parallel **Programming** :
search algorithm. This algorithm has a high **computational** complexity when applied to matching problems
often has to think in simultaneities while **construct**ing a program, because she or he often has to
ftp.dai.ed.ac.uk/pub/daidb/papers/rp722.ps.gz

Diets for Fat Sets - Erwig (1993) (Correct)
J. Functional **Programming** 1 (1)1-000, January 1993 c fl 1993
All the k-subsets S that contain x and y can be **construct**ed by fixing x and y and choosing the missing k
sense of "the same amount of information with less **nodes**"In the next section we define the discrete
voss.fernuni-hagen.de/import/pi4/erwig/papers/Diet_JFP98.ps.gz

A New Programming Paradigm for Engineering Design Software - Salustri, Venter (1994) (Correct)
A New **Programming** Paradigm for Engineering Design Software F.

aspects of functional requirements and design **intent**) effectively and efficiently. 1 Introduction
conceptual model of the engineering domain to a **computational** model will lead to new **programming** paradigms
salustri.esxf.uwindsor.ca/~fil/Papers/designer-ewc94.ps

Concurrent Object-Oriented Programming Using Term Graph.. - George Papadopoulos (1996) (Correct)
1996 (to appear)1 Concurrent object-oriented **programming** using term graph rewriting techniques George A
Fax: 357-2-339062. The generalised **computational** model of Term Graph Rewriting Systems is used
exist (as, for instance, in the case where t is a **constructor**) notification takes place: the active marking
www.kypros.org/UCY/ucy/cs/IST.ps.gz

epsilon-Transformation: Exploiting Phase Transitions to.. - Zhang, Pemberton (1994) (Correct) (3 citations)
Intern. Colloquium on Automata, Languages and **Programming**, England, July 16-20 1990. 13] P.C.
to above that point. The earliest evidence of **computational** phase transitions was the phase transition of
tree model, we show that the expected number of **nodes** expanded by branch-and-bound (BnB) using
ftp.cs.ucla.edu/tech-report/94-reports/940003.ps.Z

Symbolic Composition - Correnson, Duris, Parigot, Roussel (Correct)
Abstract To be modular, functional **programming** widely uses function compositions. Usually,
another one, where intermediate data-structure **construct**ions have been eliminated. The first approach
head l)nil !l let flat t l =case t with **node** left right !flat right (flat left l)leaf n !
pauillac.inria.fr/cdrom/ftp/fnc2/publications/SC97.ps.gz

A Polynomial Time Algorithm for Finding Finite Unions.. - Arimura, Shinohara.. (1993) (Correct)
which is a fundamental data structure in logic **programming** and term rewriting systems. In this paper, we
[Pit89] L. Pitt. Inductive inference, DFAs, and **computational** complexity. In K. P. Jantke, editor,
]6 ?k from Proposition 12. Because we can **construct** a counter example in the case where]6 is less
www.i.kyushu-u.ac.jp/~arim/papers/nil91.ps.Z

A Data-Parallel Algorithm For Minimum-Width Tree Layout And Its.. - Yang (1995) (Correct)
in general. KeyWords: Algorithms, Parallel **Programming** correctness proof, data-parallel algorithms,
problem is to compute the coordinates of **nodes** of a **tree** so that the **tree**, when drawn on a piece
www.cis.nctu.edu.tw/~wuuyang/./papers/TREEDRAW.ps

BAYES-LIN: An object-oriented environment for Bayes linear local .. - Wilkinson (1997) (Correct)
of LISP-STAT, and the basics of object-oriented **programming**. On-line, LISP-STAT information is available
a rather low-level set of tools for a back-end **computational** engine, together with diagnostic graphics for
and so the clique-**tree** can be directly **construct**ed as follows. q0,w1,q1 q1,w2,q2 q2,w3,q3 n1,q1,
www.ncl.ac.uk/~nstat/preprints/STA97-20.ps.gz

Optimal Multicast with Packetization and Network Interface Support - Ram Kesavan (1997) (Correct) (5 citations)
protocol. Such network interfaces can be **programmed** to support efficient multicasting to eliminate
using such network interface support. A method to **construct** contention-free k-binomial **trees** on
networks also provide network interface support for **nodes**, which typically includes a coprocessor and
ftp.cis.ohio-state.edu/pub/communication/techreports/tr10-97-packet-mcast.ps.Z

Distributed Algorithms for Multicast Path Setup in Data Networks - Fred Bauer (1995) (Correct) (16 citations)
spanning **trees**,ACM Transactions on **Programming** Languages and Systems, vol. 5, no. 1, pp.
at each **node** in the network, ii) use minimal **computational** and network resources, iii) require a
tree in a point-to-point network of switch **nodes**, such as a wide-area ATM network, can be modeled
ftp.cse.ucsc.edu/pub/hsnlab/fred-globecom-95.ps.Z

Generalized Queries on Probabilistic Context-Free Grammars - Pynadath, Wellman (1996) (Correct) (5 citations)
parser, and is generated using a similar dynamic-**programming** approach. We present an algorithm for
Moreover, Bayesian networks are convenient **computational** devices, supporting the calculation of
variety of types of evidence. Our method works by **construct**ing a Bayesian network to represent the
linux.eecs.umich.edu/.5/people/wellman/aaai96.ps.Z

Highly Scalable Data Balanced Distributed B-trees - Padmashree Krishna (1995) (Correct)
Symposium on Principles and Practice of Parallel **Programming**, ACM 1989, pp. 197-206. JC92] Johnson T. and
at the performance aspects of our approach of **construct**ing distributed search structures. We develop
a distributed B-**tree** that replicates its interior **nodes**. The dE-**tree** is a dB-**tree** in which leaf **nodes**

ftp.cis.ufl.edu/cis/tech-reports/tr95/tr95-015.ps

Data Structures and Algorithms for Nearest Neighbor Search in.. - Yianilos (1993) (Correct) (53 citations)
Peter N. Yianilos Abstract We consider the **computational** problem of finding nearest neighbors in method for these difficult search problems. **Tree construct**ion executes in O(n log(n)time, and search is query sequences. Like the kd-**tree**, each vp-**tree node** cuts/divides the space. But rather than using
www.neci.nj.nec.com/homepages/pny/papers/vptree/vptree.ps

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (RI) Google (Web) CSB DBLP

Results 1 - 12 of 12

1 The family of concurrent logic programming languages
Ehud Shapiro
September 1989 ACM Computing Surveys (CSUR), Volume 21 Issue 3
Full text available: pdf(9.62 MB) Additional Information: full citation, abstract, references, c

Concurrent logic languages are high-level programming languages for parallel range of both known and novel concurrent programming techniques. Being log many advantages of the abstract logic programming model, including the logic the convenience of representing data structures with logical terms and manipu amenability to metaprogrammin ...

2 Concurrency control in advanced database applications
Naser S. Barghouti, Gail E. Kaiser
September 1991 ACM Computing Surveys (CSUR), Volume 23 Issue 3
Full text available: pdf(4.69 MB) Additional Information: full citation, references, citings, in

Keywords: advanced database applications, concurrency control, cooperative t extended transaction models, long transactions, object-oriented databases, re

3 Programming languages for distributed computing systems
Henri E. Bal, Jennifer G. Steiner, Andrew S. Tanenbaum
September 1989 ACM Computing Surveys (CSUR), Volume 21 Issue 3
Full text available: pdf(6.50 MB) Additional Information: full citation, abstract, references, citin

When distributed systems first appeared, they were programmed in traditiona addition of a few library procedures for sending and receiving messages. As di commonplace and more sophisticated, this ad hoc approach became less satis began designing new programming languages specifically for implementing dis their history, their underlying pr ...

4 DROOPI: towards a generic middleware
Thomas Quinot, Fabrice Kordon, Laurent Pautet
June 2001 ACM SIGAda Ada Letters, Volume XXI Issue 2
Full text available: pdf(1.34 MB) Additional Information: full citation, abstract, re

This paper presents our work to bridge the Ada 95 Distributed Systems Annex both environments facilities. Our project consists in two successive steps. The translator. The second one aims at the definition of a generic middleware to b propose a definition and an architecture of services for a generic middleware, customized according various cr ...

5 The Pan language-based editing system
Robert A. Ballance, Susan L. Graham, Michael L. Van de Vanter
January 1992 ACM Transactions on Software Engineering and Methodology (TOS
Full text available: pdf(2.43 MB) Additional Information: full citation, abstract, references, citing

Powerful editing systems for developing complex software documents are diffi incremental algorithms and complex data structures, such editors must accom consistent, coherent, and powerful user interface, support individual variations a sharable database of information concerning the documents being edited, an in the environme ...

Keywords: Ladle, Pan, coherent user interfaces, colander, contextual constrain abstraction, interactive programming environment, logic programming, logical syntax-recognizing editor, tolerance for errors and anomalies

6 Magic conditions
Inderpal Singh Mumick, Sheldon J. Finkelstein, Hamid Pirahesh, Raghu Ramakri
March 1996 ACM Transactions on Database Systems (TODS), Volume 21 Iss
Full text available: pdf(3.14 MB) Additional Information: full citation, abstract, references, citing

Much recent work has focused on the bottom-up evaluation of Datalog program One approach, called magic-sets, is based on rewriting a logic program so that program avoids generation of irrelevant facts [Bancilhon et al. 1986; Beeri and 1991]. It was widely believed for some time that the principal application of th computation in recursive queries usin ...

Keywords: Starburst, bottom-up evaluation, constraint logic programming, con query optimization, relational databases

7 Storing a sparse table
Robert Endre Tarjan, Andrew Chi-Chih Yao
November 1979 Communications of the ACM, Volume 22 Issue 11
Full text available: pdf(594.59 KB) Additional Information: full citation, abstract, refe

The problem of storing and searching large sparse tables is ubiquitous in comp storing such tables is hashing, but hashing has poor worst-case performance. storing a static table of n entries, each an integer between 0 and N - 1. The m allows O(logn N) ...

Keywords: Gaussian elimination, parsing, searching, sparse matrix, table com

8 A Status Report on Computing Algorithms for Mathematical Programming
William W. White
September 1973 ACM Computing Surveys (CSUR), Volume 5 Issue 3
Full text available: pdf(3.02 MB) Additional Information: full citation, references, citings, index terms

9 Generators in Icon
Ralph E. Griswold, David R. Hanson, John T. Korb
April 1981 ACM Transactions on Programming Languages and Systems (TOPLAS),
Full text available: pdf(1.03 MB) Additional Information: full citation, references, citings, index term

10 Beyond structured programming
S. Pan, R. G. Dromey
May 1996 Proceedings of the 18th international conference on Software engineeri
Full text available: pdf(953.02 KB) Additional Information: full citation, references, index terms

11 Aspects of applicative programming for file systems (Preliminary Version)
Daniel P. Friedman, David S. Wise
March 1977 Proceedings of an ACM conference on Language design for reliable sof
Full text available: pdf(1.47 MB) Additional Information: full citation, abstract, references, c

This paper develops the implications of recent results in semantics for applicat evaluation (call-by-need) to the arguments of file construction functions result computation and output. The programmer need not participate in the determin evaluation of his program. Problems concerning multiple input and multiple ou is illustrated with an exam ...

Keywords: Functional combination, Real time, Recursive programming, Refere Suspension, Text editor

12 Augmentation of object-oriented programming by concepts of abstract data
Walter G. Olthoff
June 1986 Conference proceedings on Object-oriented programming systems, la
Full text available: pdf(1.34 MB) Additional Information: full citation, abstract, references, c

Object-oriented programming and abstract data type (ADT) theory have emerg science: the inability to deal efficiently with 'programming in the large' during approaches has led to significant practical and theoretical results resp. Neverth now the mutual influence seems to be limited to more or less syntactical issue forms). In ...

Results 1 - 12 of 12

Your search matched **9** of **951805** documents.

A maximum of **9** results are displayed, **15** to a page, sorted by **Relevance** in **descending** order.
You may refine your search by editing the current search expression or entering a new one the text box.

Then click **Search Again**.

(computational construct)

Search Again

Results:
Journal or Magazine = **JNL** Conference = **CNF** Standard = **STD**

1 **Stochastic dynamic programming: advanced computing constructs**
Hanson, F.B.;
Decision and Control, 1989., Proceedings of the 28th IEEE Conference on , 13-15 Dec. 1989
Page(s): 901 -903 vol.1

[Abstract] [PDF Full-Text (244 KB)] **IEEE CNF**

2 **Constructions for difference triangle sets**
Yeow Meng Chee; Colbourn, C.J.;
Information Theory, IEEE Transactions on , Volume: 43 Issue: 4 , July 1997
Page(s): 1346 -1349

[Abstract] [PDF Full-Text (184 KB)] **IEEE JNL**

3 **Fast rotation of volume data on parallel architectures**
Schroder, P.; Salem, J.B.;
Visualization, 1991. Visualization '91, Proceedings., IEEE Conference on , 22-25 Oct. 1991
Page(s): 50 -57, 409

[Abstract] [PDF Full-Text (748 KB)] **IEEE CNF**

4 **Robot skill learning, basis functions, and control regimes**
Schneider, J.G.; Brown, C.M.;

Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on , 2-6 May 1993
Page(s): 403 -410 vol.1

[Abstract] [PDF Full-Text (624 KB)] **IEEE CNF**

5 **Constructs for building complex symbolic-connectionist systems**
Khosla, R.; Dillon, T.;
Tools with Artificial Intelligence, 1994. Proceedings., Sixth International Conference on , 6-9 Nov. 1994
Page(s): 796 -799

[Abstract] [PDF Full-Text (244 KB)] **IEEE CNF**

6 **An artificial nerve fiber for evaluation of nerve cuff electrodes**
Andreasen, L.N.S.; Struijk, J.J.; Haugland, M.;
Engineering in Medicine and Biology society, 1997. Proceedings of the 19th Annual International Conference of the IEEE , Volume: 5 , 30 Oct.-2 Nov. 1997
Page(s): 1997 -1999 vol.5

[Abstract] [PDF Full-Text (208 KB)] **IEEE CNF**

7 **Combining constraints and data-flow in a visual query language**
Chavda, M.; Wood, P.T.;
Visual Languages, 1997. Proceedings. 1997 IEEE Symposium on , 23-26 Sept. 1997
Page(s): 125 -126

[Abstract] [PDF Full-Text (216 KB)] **IEEE CNF**

8 **Fuzzy adaptive logic networks**
Pedrycz, W.; Pizzi, N.J.;
Fuzzy Information Processing Society, 2002. Proceedings. NAFIPS. 2002 Annual Meeting of the North American , 27-29 June 2002
Page(s): 500 -505

[Abstract] [PDF Full-Text (602 KB)] **IEEE CNF**

9 **High-speed non-linear asynchronous pipelines**
Ozdag, R.O.; Singh, M.; Beerel, P.A.; Nowick, S.M.;

Design, Automation and Test in Europe Conference and Exhibition, 2002. Proceedings , 4-8 March 2002
Page(s): 1000 -1007

[Abstract] [PDF Full-Text (437 KB)] **IEEE CNF**